Sissejuhatus infotehnoloogiasse

Loengu sisu

- Ülevaade kursuse eesmärkidest
- Toimumisajad, lektor
- Kirjandus, lisamaterjalid, viited
- Loengukava
- Erinevad koolkonnad
- Põhimõisteid

Kursuse eesmärk

- Anda kokkusurutud ülevaade infotehnoloogiast tervikuna.
- Anda ajalooline ülevaade IT teooria, tehnoloogia ja äri arengust
- Tutvustada edaspidiseid loengukursusi stuudiumi vältel.

NB! Kursus EI ÕPETA elementaarset arvutikasutamist.

Mida kursus annab

Võimaldab esimese semestri jooksul enamvähem aru saada:

- mis on infotehnoloogia, mis on meie eriala
- millest edaspidistes IT kursustes juttu tuleb
- kuidas erinevad IT teemad omavahel seotud on
- mis teema on milleks oluline: praktikas ja teoorias
- kuidas IT on siiamaani arenenud ja mida võiks lähiaastatel oodata
- mis on praegused kuumad teemad IT-tehnoloogias ja äris

Kursuse aeg, koht, arvestus, praktika

- Kokku 16 loengut
- Praktikumid eraldi
- Kursus lõpeb kirjaliku eksamiga

Kursuse materjalid ja tööd

- Kursusel **EI OLE** ühtegi konkreetset õpikut.
- Suur osa materjale ilmub loengute eel või järel (reeglina samal päeval) **võrku**. Alati on olemas loengu üldplaan ja põhipunktid, kuid palju seletavaid detaile on võrgus puudu.

http://www.lambda.ee/index/itv0010

- Loengumaterjale on nii eesti- kui ingliskeelseid.
- Lisaks otsestele loengumaterjalidele on võrgus antud ka muud loengu teema omandamiseks vajalikud materjalid, enamasti ingliskeelsed.
- Seletavaid detaile saab kuulda ainult loengus käies. Iseõppimine on teoreetiliselt võimalik, kuid on raske ja ajamahukas.
- Kursuse praktiliste töödena (arvutil) kasutame peamiselt David Ecki praktikumide osi ja tarkvara.
- Kursuse muud kohustuslikud tööd on lihtsalt lugemisülesanded.

Esialgu soovitav kirjandus

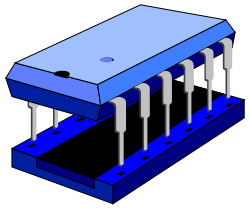
- **Kirjandus:**
  - Paul Grahami esseed.
  - New Hacker’s Dictionary.
  - Loogika: mõtlemisest tõestamiseni.
  - The Cathedral and the Bazaar
  - Gnu manifestid
- **Soovitatavad igapäevaselt loetavad lingid:**
  - www.news.com
  - www.slashdot.org
  - www.digg.com või www.reddit.com
  - ... vaata kursuse saidilt juurde!

Kursuse *esialgne* loengukava

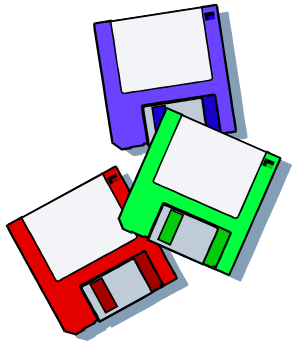
Vaata: www.lambda.ee/index/itv0010

Riistvara ja tarkvara

- **Riistvara**
  - Füüsilised, käegakatsutavad osad.
  - Näited: klaviatuur, monitor, integraalskeemid jms



- **Tarkvara**
  - Programmid ja andmed
  - Programm on jada instruktsioone



Algoritm ja programm

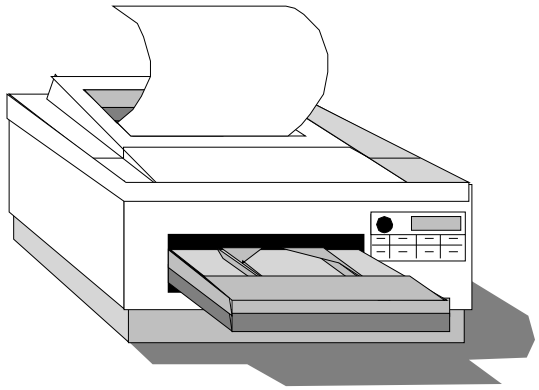
- **Algoritm** on täpne samm-sammuline, kuid mitte tingimata formaalne juhend millegi tegemiseks. Näited:
  - Toiduretsept.
  - Juhend ruutvõrrandi lahendamiseks.
- **Algoritmiline probleem -** probleem, mille lahenduse saab kirja panna täidetavate juhendite loeteluna.
- **Programm** on formaalses, üheselt mõistetavas keeles kirja pandud algoritm. Arvutid suudavad täita ainult programme.

Analoog- ja digitaalsüsteem

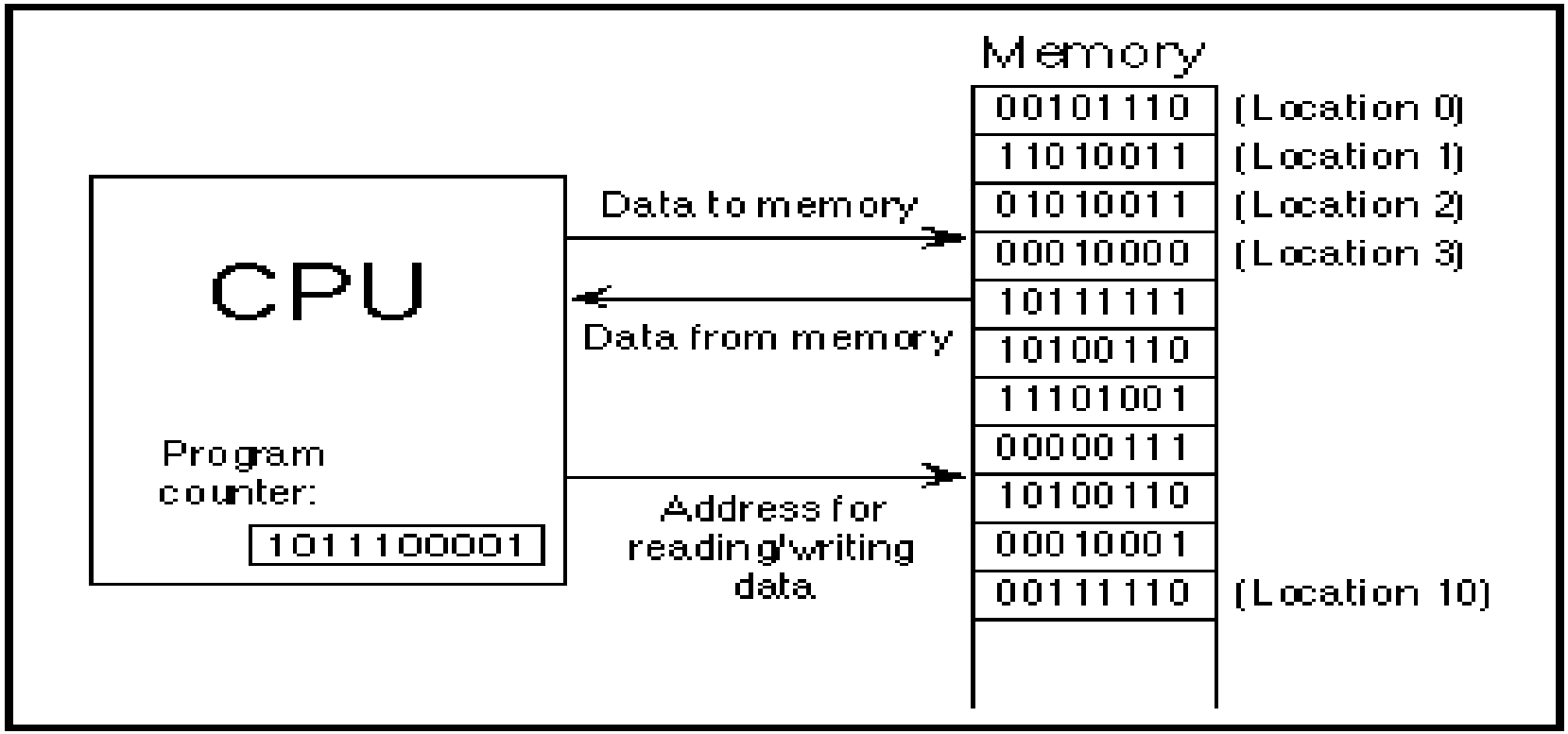
- **Analoogsüsteem**
  - andmeid salvestatakse (peegeldatakse) proportsionaalselt
  - Näit: termomeeter, vinüülplaat, foto
- **Digitaalsüsteem**
  - (pidevad) andmed lõhutakse üksikuteks tükkideks, mis salvestatakse eraldi
  - Näit: CD, arvutiprogramm, kiri tähtede ja bittidena
- **Ühelt teisele: digitaliseerimine**

Arvuti ehitus

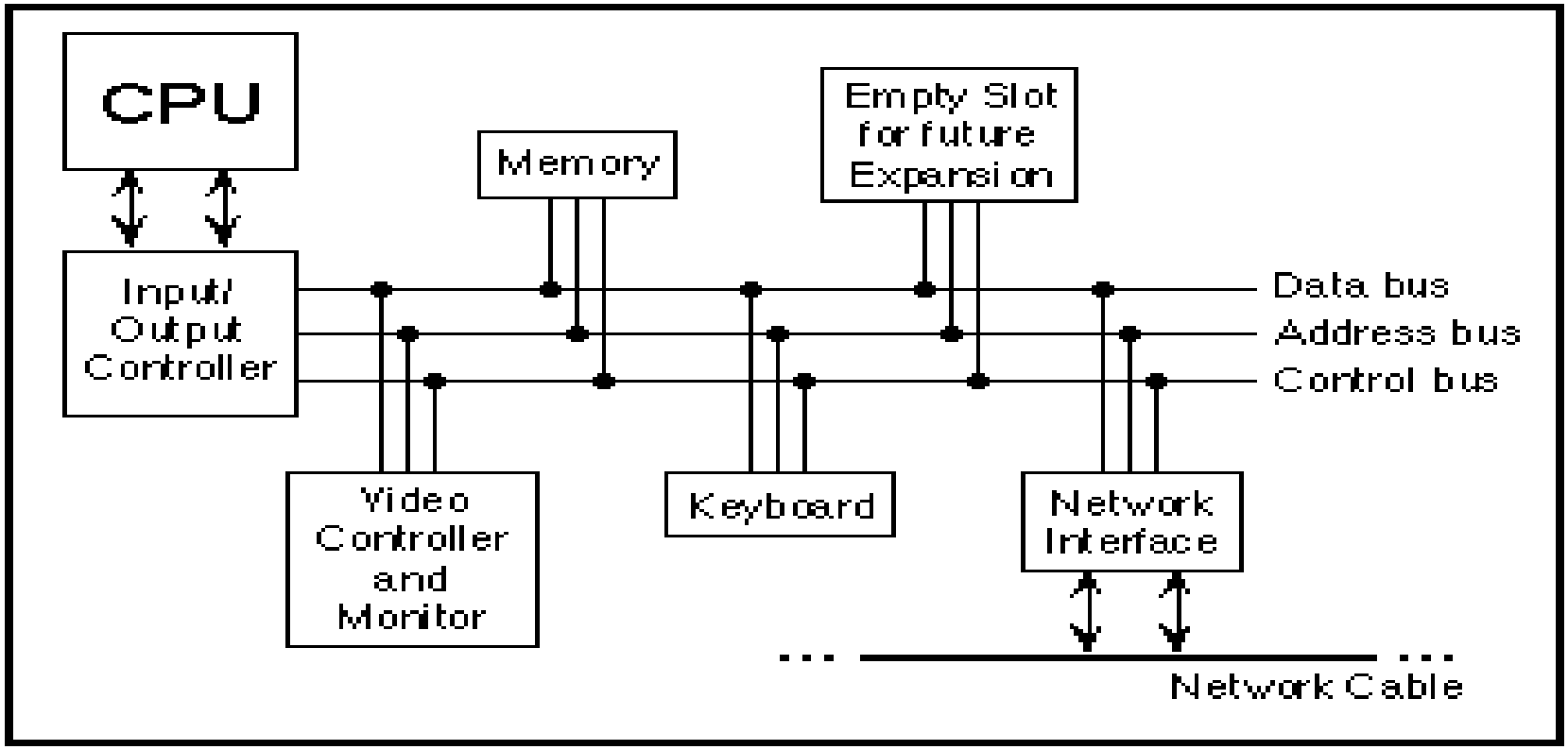
- **Põhiprotsessor** - teeb pea kogu töö
- **Põhimälu** - hoiab aktiivses kasutuses olevaid programme ja andmeid
- **Välismälu** - pikaajaliseks säilitamiseks (kõvaketas, flopid jne)
- **Välisseadmed** - monitor, klaviatuur jne



Põhiprotsessor (CPU) ja mälu



CPU ja muud seadmed



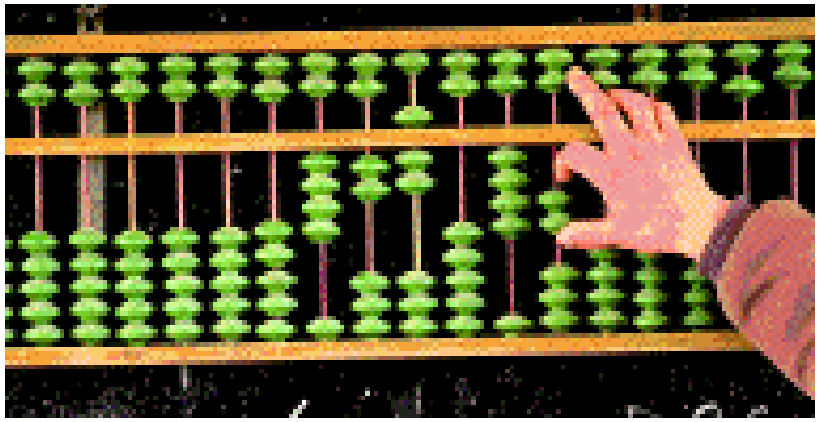
What Is (and Isn't) a Computer

- A **computer** is a device which takes data in one form, uses it, and produces a different form of information which is related to (but not the same as) the original data.

Univ. of Virginia

What Is (and Isn't) a Computer

- The Abacus is not a computer by our definition.
- It is an early calculation device that only holds numbers for the person using it.



Univ. of Virginia

What Is (and Isn't) a Computer

- Stonehenge is a computer by our definition.
- It takes the movement of the planets, sun and other heavenly bodies and provides information concerning eclipses and other astronomical events.

Univ. of Virginia

What Is (and Isn't) a Computer

- The bathroom scale is a computer by our definition.
- It takes in the amount of gravitational pull between a human body and the earth and provides us with the amount of pounds or kilograms.

Univ. of Virginia

What Is (and Isn't) a Computer

- A calculator is a computer by our definition.
- They range from doing simple arithmetic to powerful models that produce graphic output.

The Many Kinds of Computers

- **The three major comparisons of computers are:**
  - Electronic computers versus Mechanical computers
  - General-purpose versus Special-purpose computers
  - Digital versus Analog computers

The Many Kinds of Computers

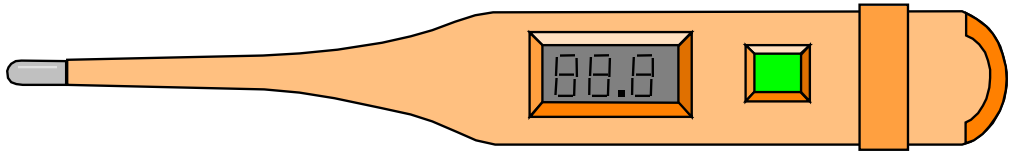
- **Electronic Computers**
  - Constructed from transistors that use electricity to function.
- **Mechanical Computers**
  - Do not use electricity to function.
  - Constructed of a combination of gears, levers and springs.

The Many Kinds of Computers

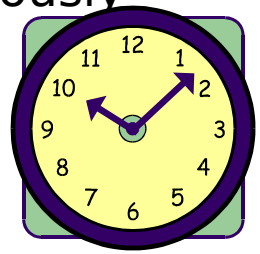
- **General-purpose Computers**
  - Were not manufactured to do any one thing.
  - Changeable to do any task.
- **Special-purpose Computers**
  - Manufactured to do a predetermined task or set of tasks.

The Many Kinds of Computers

- **Digital Computers**
  - One that functions in discretely varying quantities.
  - Produces or gives results that are also discretely varying.



- **Analog Computers**
  - One that functions in continuously varying quantities.
  - Produces or gives results that are also continuously varying.



The General-Purpose Digital Computer

- **The General-Purpose Digital Computer**
  - Accepts information of many kinds.
  - Changes it in a way that is controlled by humans.
  - Presents results in a way usable by humans.

What Is Information?

- **The five types of information that are the only types the computer commonly manipulates:**
  - Visual (pictures)
  - Numeric (numbers)
  - Character (text)
  - Audio (sound)
  - Instructions (programs)

What Is Information?

- **Before the computer can use any type of information, it must be stored in the computer's memory.**
- **Problem: How is information stored within the computer?**
  - Information is stored in numerical form within the computer
  - Modern computers work in a system of numbers called binary numbers

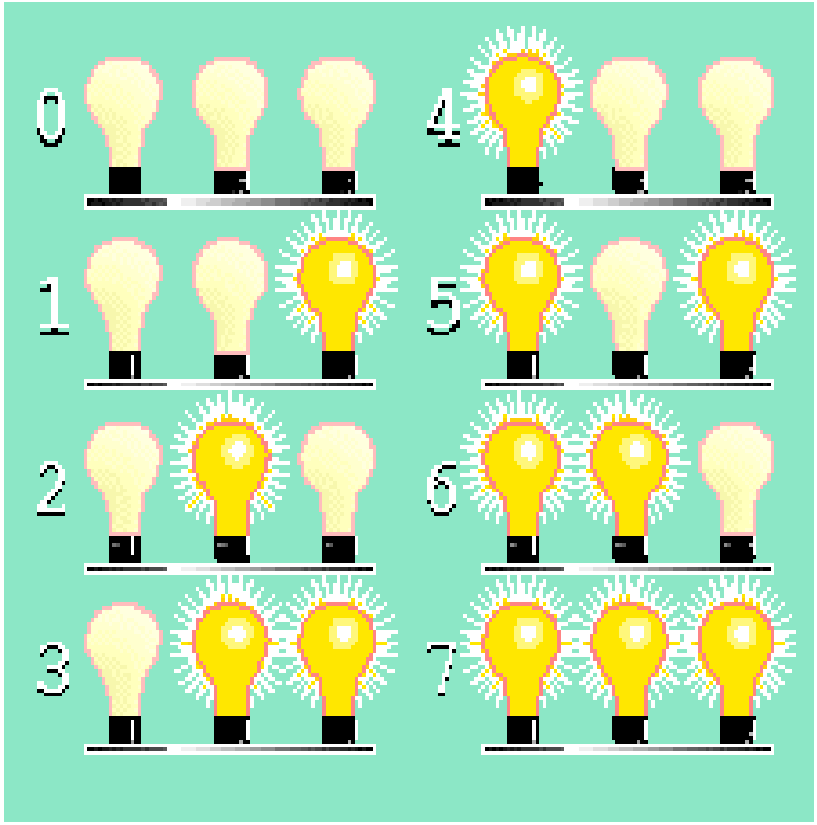
What Is Information?

- Binary numbers:
  - Similar to familiar decimal system.
  - Uses only two symbols: 0 and 1.
  - The choice of using binary numbers is dictated by cost and reliability.
- Binary circuits:
  - Electronic circuits are cheapest and most reliable if they only assume two states or conditions.
  - These binary circuits have only two states, ON or OFF.

Univ. of Virginia

Representation of Numbers

- Binary numbers use only two symbols: 0 and 1.
- How can more than two possibilities be represented?
- A three light system can have up to eight combinations. Each combination can represent a code.



Representation of Numbers

- **Binary equivalents of the numerals 0 to 7.**

0	000
1	001
2	010
3	011
4	100
5	101
6	110
7	111

Binary numerals

Representation of Numbers

- **The** decimal **system:**

$$
\begin{array}{lcccl}
 & & \begin{array}{ccc} 100\text{s} & 10\text{s} & 1\text{s} \\ \downarrow & \downarrow & \downarrow \end{array} & & \\
312 & = & 3 \quad 1 \quad 2 & = & (3\text{x}\ 10^2) + (1\text{x}10^1) + (2\text{x}10^0) \\
 & & & = & (3\text{x}100) + (1\text{x}10) + (2\text{x}1) \\
 & & & = & 300 + 10 + 2
\end{array}
$$

Representation of Numbers

- **The** binary **system:**

$$
\begin{array}{rccc}
 & 4s & 2s & 1s \\
 & \downarrow & \downarrow & \downarrow \\
101_{\text{binary}} = & 1 & 0 & 1
\end{array}
\begin{aligned}
&= (1 \times 2^2) + (0 \times 2^1) + (1 \times 2^0) \\
&= (1 \times 4) + (0 \times 2) + (1 \times 1) \\
&= 4 + 0 + 1 \\
&= 5
\end{aligned}
$$

$$101_{\text{binary}} = 5 \text{ in decimal}$$

Representation of Numbers

- **Translate 10011 $_{\text{binary}}$ into a decimal number.**

1. Place the base two numerals under the place values.

2^4	2^3	2^2	2^1	2^0
16	8	4	2	1
1	0	0	1	1

2. Multiply through by the place values.

2^4	2^3	2^2	2^1	2^0
16	8	4	2	1
1	0	0	1	1
1x16	*0x8*	*0x4*	*1x2*	*1x1*

3. Add up the column for the decimal number.

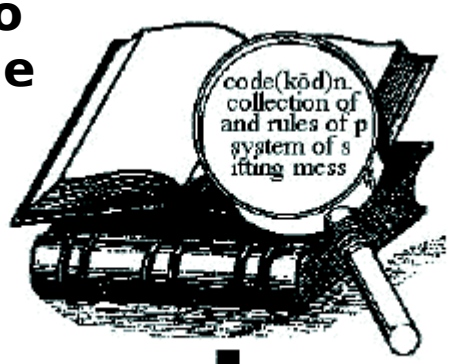
1
2
0
0
16
19

$10011_{\text{binary}} = 19$ in decimal

Representing Symbols and Text

- **Each letter and symbol in a text document must be translated into a binary number for storage in the computer.**

- **Standardized means of storing these codes:**
  - **ASCII** (American Standard Code for Information Interchange)
  - **EBCDIC** (Extended Binary Coded Decimal Interchange Code)
  - **UNICODE (**Extended ASCII)



Coded in (decimal) ASCII

99 111 100 101 40 107 111 100 41 110 46
99 111 108 108 101 99 116 105 111 110 32 111 102
97 110 100 32 114 117 108 101 115 32 111 102 112
115 121 115 116 101 109 32 111 102 32 115
105 116 116 105 110 103 32 109 101 115 115

Coded in (binary) ASCII

1100011 1101111 1100100 1100101 0101000 1101011 1101111 1100100 0101001 1101110 0101110
1100011 1101111 1101100 1101100 1100101 1100011 1110100 1101001 1101111 1101110 0100000 1101111 1100110
1100001 1101110 1100100 0100000 1110010 1110101 1101100 1100101 1110011 0100000 1101111 1100110 0100000 1110000
1110011 1111001 1110011 1110100 1100101 1101101 0100000 1101111 1100110 0100000 1110011
1101001 1110100 1110100 1101001 1101110 1100111 0100000 1101101 1100101 1110011 1110011

Representing Pictures

- **Pictures must be translated into a binary format for storage in the computer.**
  - The picture is broken down into small elements.
  - These elements are called Pixels (Picture Elements).

- Digitizer:
  - A device that converts a picture into a binary format for storage in the computer.
  - Examples of digitizers: scanner, digital camera.

Univ. of Virginia

Representing Pictures

- **Digitized picture of a tiger.**

Univ. of Virginia

Representing Pictures

- **Black and white** pixels **are either 0 or 1.**

```
00010000000000000000000000000000000000000010101
01000000000000010000000000000000000000000000101
01100000000000010000000000000000000000000000011
10101010000000001000000000000000000000000000011
10010000000000001011000000000000000000000000101
01000100000000000111100000000000000000000000111
01001001000000000111110101010111000000000000011
00010000000000001111011101111111010000000001011
00001010010000011011111011111101100000000001111
00000101000000001111101111010111010000000000111
00000101001000111010101010110100000000000010111
00000010101000000111010101011010101000000011111
00000101100000010101001000000000000000000001110
00000000000110010101000000000000000000000001111
00000000000101100101000000000000000000000101111
00100101010101001001010100000000000000001011111
10000011111101001011101010101100000001011011 1
10010011110101111111111101011011010111111111111
01100101111101111111111111111111101111110111111
10101011011111111111111111111111111111111111111
10100001110111111111111111111111111111111111111
01010100111111111111111111111111111111111111111
01100001010111111111111111111011111100111101 01
01011110111111111111111111111111011010101110101
10101010111111111111111101011011110111011111 1
00000010111111111111010111011010011111101010 10
00000010111111111111011111111100101111011010 10
00000001111110101111101110111010011111101010 10
00000001111111111111111010111111101111110111011
00001011111101101010110000101111111111111101011
```

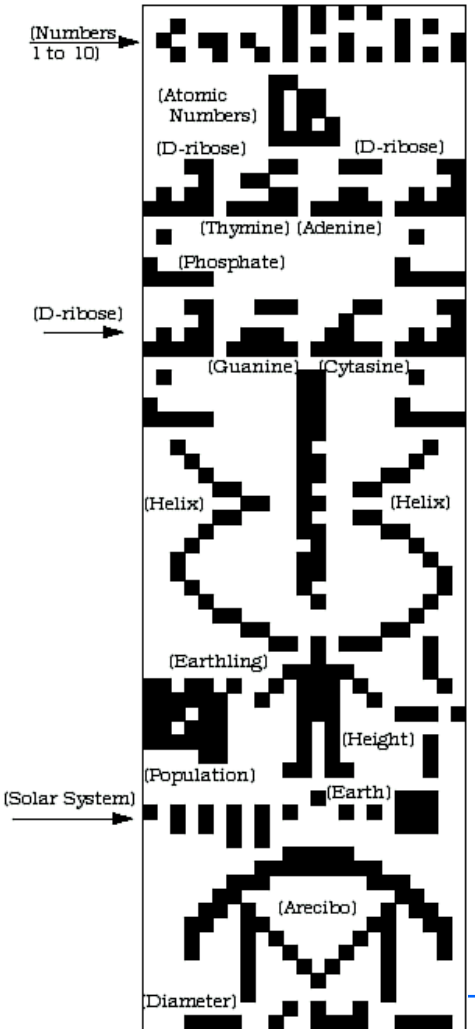
Representing Pictures

- Gray-Scale:
  - Each pixel contains a value representing some shade of gray.
  - The more shades of gray possible, the more memory will be needed.
    - 4 shades of gray needs 2 bits per pixel:
      - 00, 01, 10, 11
    - 8 shades of gray needs 3 bits per pixel:
      - 000, 001, 010, 011, 100, 101, 110, 111
    - 64 shades of gray needs 6 bits per pixel:
      - 000000, 000001, … 111110, 111111

Univ. of Virginia

Representing Pictures

- **Message transmission from the Arecibo radio telescope in Puerto Rico to other stars.**



Univ. of Virginia

Storage of Binary Information

- **Capacity**

Unit	*Description*	*Approximate Size*
1 bit	1 binary digit	
1 nibble	4 bits	
1 byte	8 bits	1 character
1 kilobyte	1,024 bytes	≈1/2 page, double spaced
1 megabyte	1,048,576 bytes 1 million bytes	≈500,000 pages
1 gigabyte	1,073,741,824 bytes 1 billion bytes	≈5 million pages
1 terabyte	1 trillion bytes	≈5 billion pages

Instructions as Numbers

- **Fact:**
  - Declarative: statement of being.
  - Imparts knowledge.

- **Instruction:**
  - Imperative: demands action.
  - Controls information or activity.

Instructions as Numbers

- **Instructions:**
  - Must be stored within the computer before use.
  - Must be stored in binary form.
  - A set of binary instructions is called a program.

The Stored-program Computer

- **Program:**
  - A collection of instructions for the computer to perform one by one.

- **Machine Language:**
  - The language of the computing machine.
  - All instructions must be in the form of binary numbers (binary code).

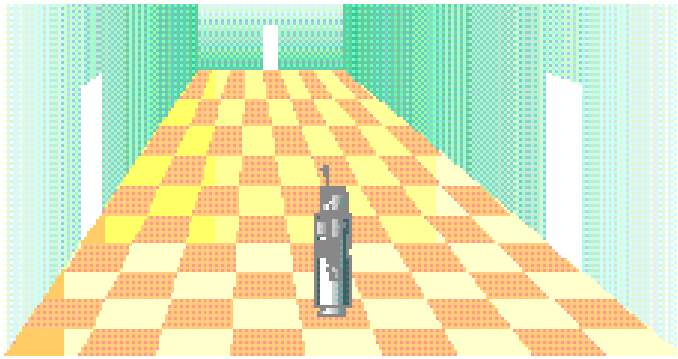
The Stored-program Computer

- **Stored-program Computer:**
  - Also known as the von Neumann-type computer.
  - Has memory - a place to keep both:
    - instructions (ie program)
    - and the needed information (ie data)

    needed for computation by the computer.

Programs and Algorithms

- **Our first example of the computer: The ROBOT computer**



- **The ROBOT's domain**
  - The room is empty.
  - The room is rectangular.
  - There may be one or more open doorways in the walls.
  - The floor is paved with square tiles with lines between them. The lines are easy to see.
  - The size of the room is unknown to us at any given time.
  - The size of the room does not change during the execution of a program.
  - Doorways will never be located in corners.

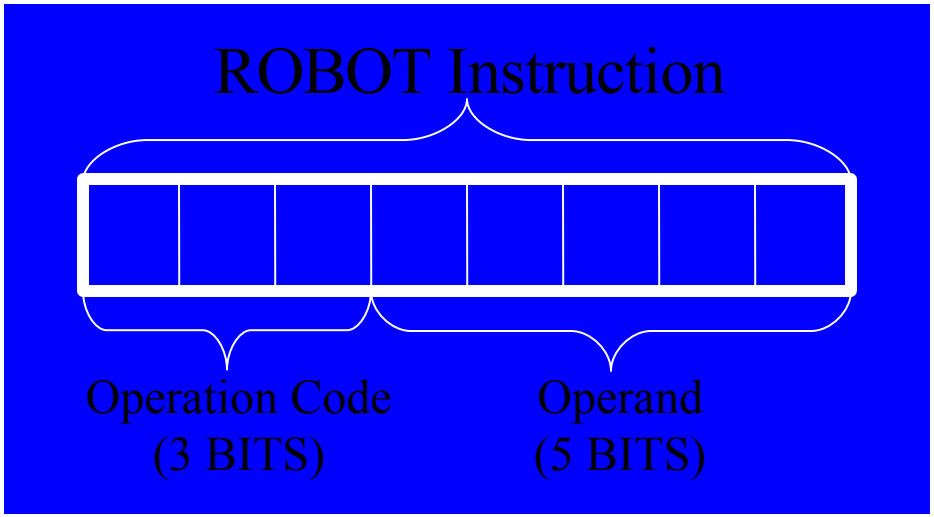
Programs and Algorithms

Simple toy programmable robot characteristics:

- **Forward movement** — Moves forward from square to square within its domain.
- **Changing direction** — Turns only 90 degrees to the right.
- **Arm movement** — Can ***raise*** and ***lower*** its arms.
- **Arm extension** — When its arms are raised, they reach to the far edge of the next square.
- **Sensors** — The sensors at the ends of its arms are used to locate walls.
- **Intelligence** — NONE. The ROBOT cannot see or think on its own. It only executed instructions stored in memory.

Programs and Algorithms

- **ROBOT instructions have two parts:**
  - **Operation Code** (Opcode) - Dictates the action to be performed by the ROBOT.
  - **Operand** (Argument) - The address of a position in memory.
  - Each part of a ROBOT instruction is called a **field**.



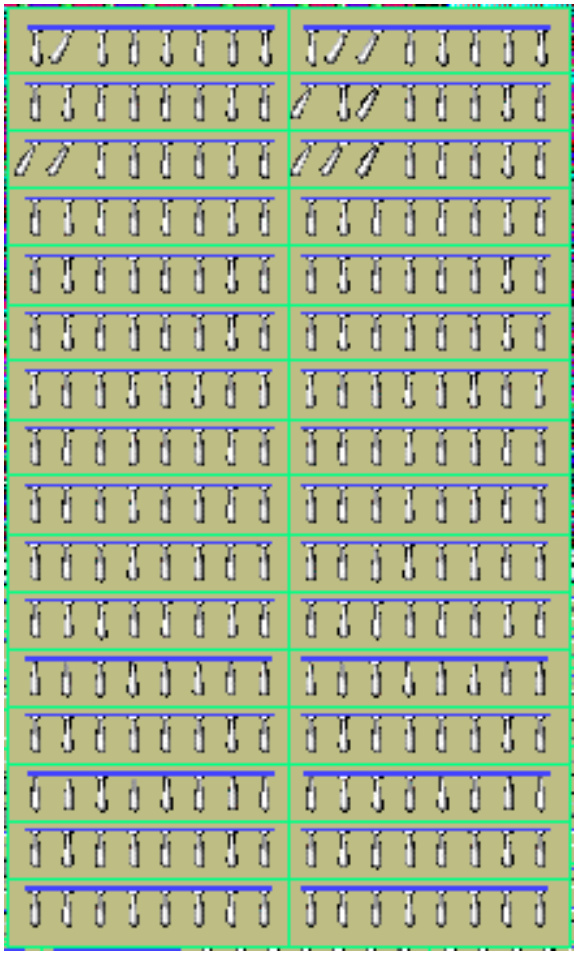
Programs and Algorithms

- **ROBOT Programs:**
  - Lists of instructions can be determined and changed by the person who operates the ROBOT.
  - **Program**: Refers to the list of instructions given to the ROBOT.
    - A program must be placed into the ROBOT's memory before any execution can take place.

- **ROBOT's Memory:**
  - Located on the ROBOT's torso.
  - 32 memory locations.
  - Each memory location is a set of 8 toggle switches.
  - On = 1 Off = 0
  - **Loading** a program: setting the switches.

Programs and Algorithms

- **The ROBOT's memory contains 32 memory locations numbered 0 to 31.**
- **Each location is capable of storing one ROBOT instruction.**



Programs and Algorithms: commands 1

Opcode	English command	Action taken by
000	**STEP**	The ROBOT takes one STEP forward if possible.
001	**TURN**	The ROBOT pivots 90 degrees to the right.
010	**RAISE**	The ROBOT raises its arms if possible. IF NOT POSSIBLE: There MUST be a wall directly in front of the ROBOT. The warning light will come on. *No other commands will be recognized until the light is turned off.*
011	**LOWER**	The ROBOT lowers its arms if they are raised.

Programs and Algorithms: commands 2

Opcode	English command	Action taken by
100	**SENSE**	The ROBOT, with its arms in *raised* position, can detect if it is one step away from the wall it is facing. IF IT IS, the warning light will turn on. *Recognizes no other commands until the light is turned off.*
101	**GOTO**	The ROBOT takes the next command out of normal order. The Operand, the last 5 bits of the instruction, tells which memory location is to be performed next.
110	**LIGHT**	*IF the light is turned on, this command turns it off.* The ROBOT will again recognize instructions in the program.
111	**STOP**	The ROBOT shuts off its own power.

Programs and Algorithms

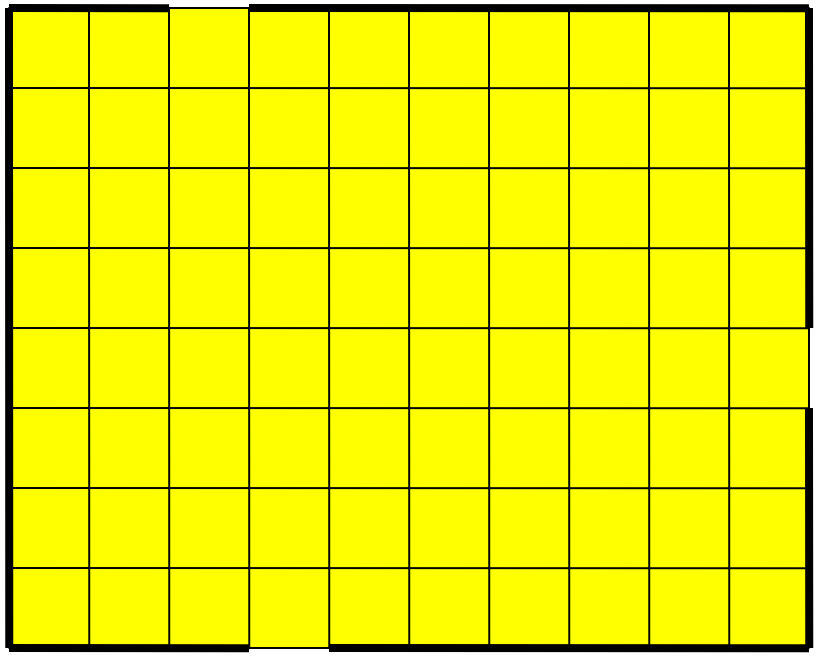
- Algorithm**:**
  - A step-by-step process used to solve a problem.
  - The general solution to the problem.
  - Usually implemented by a program.

  - Problem: Cause the ROBOT to walk to the wall it is initially facing and then stop with its arms lowered and facing against the wall. Assume the ROBOT is not initially facing an open doorway.

    - Remember:
      - We have NO IDEA how big the room is!
      - We CAN'T just tell it to STEP X-number of times!
      - The algorithm has a general solution. (Solves the problem in all situations.)

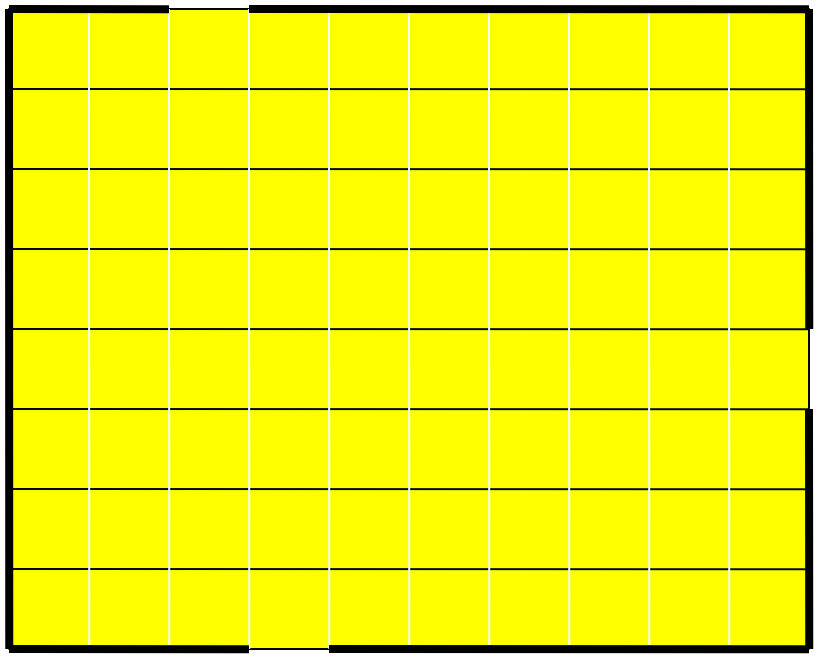
Programs and Algorithms

- **Why isn't this a "good enough" solution to the problem of finding the wall in front of the ROBOT?**
  - 0 RAISE
  - 1 SENSE
  - 2 STEP
  - 3 GOTO 1
  - 4 LIGHT
  - 5 STOP



Programs and Algorithms

- **Why is this a better solution to the problem of finding the wall in front of the ROBOT?**
  - 0 RAISE
  - 1 LOWER
  - 2 STEP
  - 3 GOTO 0
  - 4 LIGHT
  - 5 STOP



Programs and Algorithms

- **Programming the ROBOT - Taking the “English” steps and writing them in the language the ROBOT understands (Machine Language).**
  - **Machine Language** - Written in binary code, the program is in the form the computer understands.

“English” Version	Machine Language Version
RAISE	01000000
LOWER	01100000
STEP	00000000
GOTO 0	10100000
LIGHT	11000000
STOP	11100000

Programs and Algorithms

- **Loop** - A sequence of instructions which is repeated one or more times when a program is executed.
- **Infinite loop -** A set of instructions which causes the program to repeat the same commands over and over with no possible way of stopping.

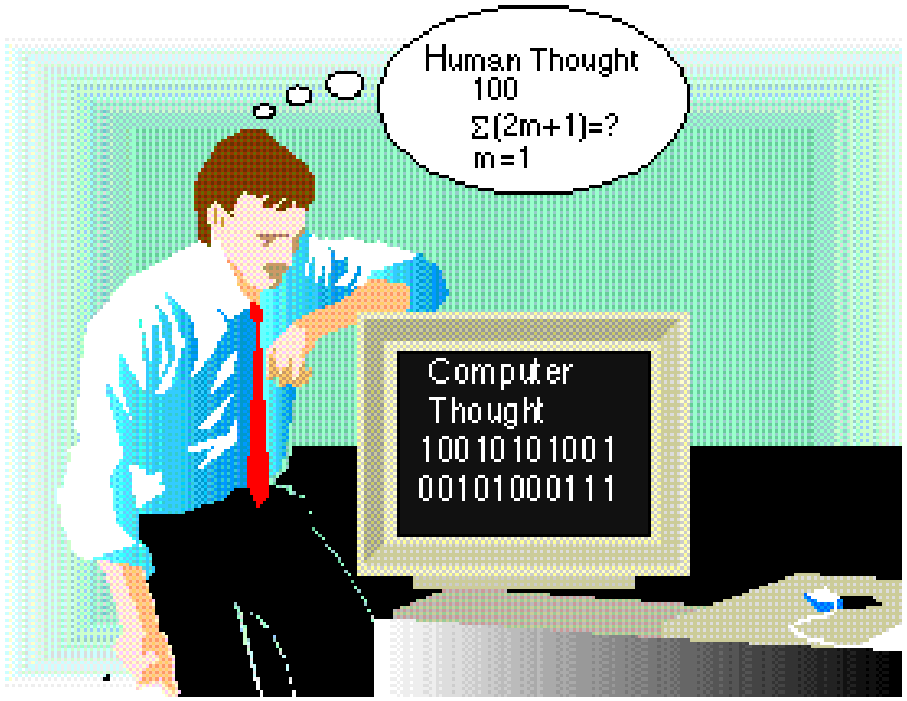
Programs and Algorithms

- **Cause the ROBOT to walk around the perimeter of the room.**
  - Does the program ever stop? What kind of loop does this program contain?

  - 0 RAISE
  - 1 LOWER
  - 2 STEP
  - 3 GOTO 0
  - 4 LIGHT
  - 5 TURN
  - 6 GOTO 0
  - 7 STOP

How does a person communicate with a computer?

- Programming languages bridge the gap between human thought process and computer binary circuitry.



Koolkonnad